WEST

Help | Logout | Interrupt

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers | Preferences | Cases

Search Results -

Terms	Documents
L1 and (memory same read$3 same writ$3 same (area or location))	26

Database: US Patents Full-Text Database; US Pre-Grant Publication Full-Text Database; JPO Abstracts Database; EPO Abstracts Database; Derwent World Patents Index; IBM Technical Disclosure Bulletins

Search: L2

Refine Search | Recall Text | Clear

Search History

DATE: Tuesday, November 18, 2003 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=USPT; PLUR=YES; OP=OR			
L2	L1 and (memory same read$3 same writ$3 same (area or location))	26	L2
L1	((universal adj1 serial adj1 bus) or USB) same host same (microcomputer or computer) same memory	208	L1

END OF SEARCH HISTORY

WEST

Help | Logout | Interrupt

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers | Preferences | Cases

Search Results -

Terms	Documents
L2	0

Database: US Patents Full-Text Database; US Pre-Grant Publication Full-Text Database; JPO Abstracts Database; EPO Abstracts Database; Derwent World Patents Index; IBM Technical Disclosure Bulletins

Search: L3

Refine Search

Recall Text | Clear

Search History

DATE: Tuesday, November 18, 2003 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=PGPB,JPAB,EPAB,DWPI,TDBD; PLUR=YES; OP=OR			
L3	L2	0	L3
DB=USPT; PLUR=YES; OP=OR			
L2	L1 and (memory same read$3 same writ$3 same (area or location))	26	L2
L1	((universal adj1 serial adj1 bus) or USB) same host same (microcomputer or computer) same memory	208	L1

END OF SEARCH HISTORY

L2: Entry 2 of 26 File: USPT Oct 14, 2003

DOCUMENT-IDENTIFIER: US 6633963 B1
TITLE: Controlling access to multiple memory zones in an isolated execution environment

Detailed Description Text (11):
One concept of the isolated execution architecture is the creation of an isolated region in the system memory, referred to as an isolated area, which is protected by both the processor and chipset in the computer system. The isolated region may also be in cache memory, protected by a translation look aside (TLB) access check. Access to this isolated region is permitted only from a front side bus (FSB) of the processor, using special bus (e.g., memory read and write) cycles, referred to as isolated read and write cycles. The special bus cycles are also used for snooping. The isolated read and write cycles are issued by the processor executing in an isolated execution mode. The isolated execution mode is initialized using a privileged instruction in the processor, combined with the processor nub loader 52. The processor nub loader 52 verifies and loads a ring-0 nub software module (e.g., processor nub 18) into the isolated area. The processor nub 18 provides hardware-related services for the isolated execution.

Detailed Description Text (25):
FIG. 1D is a diagram illustrating a computer system 100 in which one embodiment of the invention can be practiced. The computer system 100 includes a processor 110, a host bus 120, a memory controller hub (MCH) 130, a system memory 140, an input/output controller hub (ICH) 150, a non-volatile memory, or system flash, 160, a mass storage device 170, input/output devices 175, a token bus 180, a motherboard (MB) token 182, a reader 184, and a token 186. The MCH 130 may be integrated into a chipset that integrates multiple functionalities such as the isolated execution mode, host-to-peripheral bus interface, memory control. Similarly, the ICH 150 may also be integrated into a chipset together or separate from the MCH 130 to perform I/O functions. For clarity, not all the peripheral buses are shown. It is contemplated that the system 100 may also include peripheral buses such as Peripheral Component Interconnect (PCI), accelerated graphics port (AGP), Industry Standard Architecture (ISA) bus, and Universal Serial Bus (USB), etc.

Detailed Description Text (29):
The host bus 120 provides interface signals to allow the processor 110 or processors 110, 110a, and 110b to communicate with other processors or devices, e.g., the MCH 130. In addition to normal mode, the host bus 120 provides an isolated access bus mode with corresponding interface signals for memory read and write cycles when the processor 110 is configured in the isolated execution mode. The isolated access bus mode is asserted on memory accesses initiated while the processor 110 is in the isolated execution mode. The isolated access bus mode is also asserted on instruction pre-fetch and cache write-back cycles if the address is within the isolated area address range and the processor 110 is initialized in the isolated execution mode. The processor 110 responds to snoop cycles to a cached address within the isolated area address range if the isolated access bus cycle is asserted and the processor 110 is initialized into the isolated execution mode.

Detailed Description Text (30):
The MCH 130 provides control and configuration of memory and input/output devices such as the system memory 140 and the ICH 150. The MCH 130 provides interface circuits to recognize and service isolated access assertions on memory reference bus

cycles, including isolated memory read and write cycles. In addition, the MCH 130 has memory range registers (e.g., base and length registers) to represent the isolated area in the system memory 140. Once configured, the MCH 130 aborts any access to the isolated area that does not have the isolated access bus mode asserted.

Detailed Description Text (33):
The isolated bus cycle interface 152 includes circuitry to interface to the isolated bus cycle signals to recognize and service isolated bus cycles, such as the isolated read and write bus cycles. The processor nub loader 52, as shown in FIG. 1A, includes a processor nub loader code and its digest (e.g., hash) value. The processor nub loader 52 is invoked by execution of an appropriate isolated instruction (e.g., Iso-Init) and is transferred to the isolated area 70. From the isolated area 80, the processor nub loader 52 copies the processor nub 18 from the system flash (e.g., the processor nub code 18 in non-volatile memory 160) into the isolated area 70, verifies and logs its integrity, and manages a symmetric key used to protect the processor nub's secrets. In one embodiment, the processor nub loader 52 is implemented in read only memory (ROM). For security purposes, the processor nub loader 52 is unchanging, tamper-resistant and non-substitutable. The digest memory 154, typically implemented in RAM, stores the digest (e.g., hash) values of the loaded processor nub 18, the operating system nub 16, and any other critical modules (e.g., ring-0 modules) loaded into the isolated execution space. The cryptographic key storage 155 holds a symmetric encryption/decryption key that is unique for the platform of the system 100. In one embodiment, the cryptographic key storage 155 includes internal fuses that are programmed at manufacturing. Alternatively, the cryptographic key storage 155 may also be created with a random number generator and a strap of a pin. The isolated execution logical processing manager 156 manages the operation of logical processors operating in isolated execution mode. In one embodiment, the isolated execution logical processing manager 156 includes a logical processor count register that tracks the number of logical processors participating in the isolated execution mode. The token bus interface 159 interfaces to the token bus 180. A combination of the processor nub loader digest, the processor nub digest, the operating system nub digest, and optionally additional digests, represents the overall isolated execution digest, referred to as an isolated digest. The isolated digest is a fingerprint identifying the ring-0 code controlling the isolated execution configuration and operation. The isolated digest is used to attest or prove the state of the current isolated execution.

Detailed Description Text (48):
The isolated subsystem memory range setting register 280 stores the isolated subsystem memory range settings 281. The isolated subsystem memory range settings 281 can be represented as a table having a plurality of rows and columns that store the settings for the subsystems within the isolated area. This table may represent the settings for all the possible subsystems in the isolated memory area, or it may represent a cache of settings for selected subsystems. Each row corresponds to a different subsystem and has a number of column entries that describe the subsystem and the subsystem's associated memory zone (75.sub.0 -75.sub.N) in the isolated memory area 70 (FIG. 1C). Each row includes an ID value 283, an entry attribute 285, a subsystem range setting 287, and an entry context 288. The ID value is a unique identifier for each subsystem. The entry attribute 285 includes a plurality of values including page size (such as large or normal sized), read or write access, and a master subsystem flag.

WEST

Generate Collection | Print

L2: Entry 2 of 26 File: USPT Oct 14, 2003

US-PAT-NO: 6633963
DOCUMENT-IDENTIFIER: US 6633963 B1

TITLE: Controlling access to multiple memory zones in an isolated execution environment

DATE-ISSUED: October 14, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Ellison; Carl M.	Portland	OR		
Golliver; Roger A.	Beaverton	OR		
Herbert; Howard C.	Phoenix	AZ		
Lin; Derrick C.	San Mateo	CA		
McKeen; Francis X.	Portland	OR		
Neiger; Gilbert	Portland	OR		
Reneris; Ken	Wilbraham	MA		
Sutton; James A.	Portland	OR		
Thakkar; Shreekant S.	Portland	OR		
Mittal; Millind	Palo Alto	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Intel Corporation	Santa Clara	CA			02

APPL-NO: 09/ 618489 [PALM]
DATE FILED: July 18, 2000

PARENT-CASE:
RELATED APPLICATION This application claims the benefit U.S. Provisional Patent Application No. 60/198,225 filed on Mar. 31, 2000.

INT-CL: [07] G06 F 12/06, G06 F 12/14

US-CL-ISSUED: 711/163; 711/153, 711/152, 711/170, 711/173
US-CL-CURRENT: 711/163; 711/152, 711/153, 711/170, 711/173

FIELD-OF-SEARCH: 711/1, 711/147, 711/151, 711/152, 711/153, 711/163, 711/164, 711/170, 711/173, 711/220, 709/213

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL

☐	4037214	July 1977	Birney et al.	
☐	4278837	July 1981	Best	
☐	4521852	June 1985	Guttag	
☐	5022077	June 1991	Bealkowski et al.	
☐	5079737	January 1992	Hackbarth	711/164
☐	5255379	October 1993	Melo	
☐	5293424	March 1994	Holtey et al.	
☐	5386552	January 1995	Garney	
☐	5421006	May 1995	Jablon et al.	
☐	5459869	October 1995	Spilo	
☐	5473692	December 1995	Davis	
☐	5479509	December 1995	Ugon	
☐	5568552	October 1996	Davis	
☐	5615263	March 1997	Takahashi	
☐	5628022	May 1997	Ueno et al.	
☐	5657445	August 1997	Pearce	
☐	5717903	February 1998	Bonola	
☐	5729760	March 1998	Poisner	
☐	5737760	April 1998	Grimmer et al.	711/163
☐	5757919	May 1998	Herbert et al.	
☐	5764969	June 1998	Kahle et al.	
☐	5796845	August 1998	Serikawa et al.	
☐	5805712	September 1998	Davis	
☐	5835594	November 1998	Albrecht et al.	
☐	5844986	December 1998	Davis	
☐	5852717	December 1998	Bhide et al.	
☐	5872994	February 1999	Akiyama et al.	
☐	5937063	August 1999	Davis	
☐	5978481	November 1999	Ganesan et al.	
☐	6014745	January 2000	Ashe	
☐	6055637	April 2000	Hudson et al.	
☐	6058478	May 2000	Davis	
☐	6085296	July 2000	Karkhanis et al.	
☐	6125430	September 2000	Noel et al.	
☐	6148379	November 2000	Schimmel	
☐	6158546	December 2000	Hanson et al.	
☐	6175925	January 2001	Nardone et al.	
☐	6178509	January 2001	Nardone et al.	

☐	6192455	February 2001	Bogin et al.	
☐	6205550	March 2001	Nardone et al.	
☐	6249872	June 2001	Wildgrube et al.	713/200
☐	6272533	August 2001	Browne	709/213
☐	6282651	August 2001	Ashe	
☐	6282657	August 2001	Kaplan et al.	
☐	6292874	September 2001	Barnett	711/153
☐	6301646	October 2001	Hostetter	
☐	6321314	November 2001	Van Dyke	
☐	6339815	January 2002	Feng et al.	
☐	6339816	January 2002	Bausch	
☐	6499123	December 2002	McFarland et al.	
☐	6505279	January 2003	Phillips et al.	
☐	2001/0021969	September 2001	Burger et al.	
☐	2001/0027527	October 2001	Khidekel et al.	

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY	US-CL
1146715	October 2001	EP	
WO 97/29567	August 1997	WO	
WO9844402	October 1998	WO	
WO9905600	February 1999	WO	
WO9957863	November 1999	WO	
WO0062232	October 2000	WO	
WO 01/27723	April 2001	WO	
WO 01/27821	April 2001	WO	
WO 01/75565	October 2001	WO	
WO 01/75595	October 2001	WO	

ART-UNIT: 2186

PRIMARY-EXAMINER: Peikari; B. James

ATTY-AGENT-FIRM: Blakely, Sokoloff, Taylor & Zafman LLP

ABSTRACT:

A processor having a normal execution mode and an isolated execution mode generates an access transaction. The access transaction is configured using a configuration storage that stores configuration settings. The configuration settings include a plurality of subsystem memory range settings defining memory zones. The access transaction also includes access information. A multi-memory zone access checking circuit, coupled to the configuration storage, checks the access transaction using at least one of the configuration settings and the access information. The multi-memory zone access checking circuit generates an access grant signal if the access transaction is valid.

44 Claims, 8 Drawing figures

WEST

Generate Collection Print

L2: Entry 22 of 26 File: USPT Jun 6, 2000

DOCUMENT-IDENTIFIER: US 6073205 A
TITLE: System and method of write posting in a universal serial bus system

Abstract Text (1):
An apparatus and method for write posting in a universal serial bus (USB) system includes a host computer connected to USB devices via a USB. The host computer generates requests to write data to memory within the USB device. The host computer includes a queue for posting the write requests on generation thereof. The write requests are posted in the queue until the host computer transmits a single data packet generated from the posted write requests. The Data packet is generated in response to the host computer generating a request to read data from the USB device, the host computer determining that the most recently posted write request is directed to a memory location within the USB device which is nonpostable, or an indication that the queue lacks storage space for subsequent write requests. The USB device receives the transmitted Data packet from the host computer and writes data to internal memory locations in accordance with the received Data packet.

Brief Summary Text (24):
The present invention solves the aforementioned problems and others and provides an apparatus and method for posting write requests in a USB-based system. The present invention is implemented in the USB system in which a host computer is coupled to USB devices via a USB. The host computer is configured to generate requests to write data to memory of one of the USB devices at designated memory addressed therein. The host computer includes a queue for posting each write request upon generation thereof. The write requests are posted until the host computer generates a flush command in response to (1)the host computer generating a read data request, (2) the host computer determining that the most recently posted write request is directed to a USB device memory location which is nonpostable, or (3) the host computer determining that the queue lacks sufficient storage space for subsequent write requests. In response to the flush command, the host computer generates a data packet from the posted write requests. The data packet is transmitted to the USB device via the USB. The USB device, in turn, writes data to internal memory locations in accordance with the received data packets.

Brief Summary Text (25):
By posting the write requests in the queue, the host computer initiates a single bus transaction for a plurality of write requests rather than initiating a plurality of bus transactions, each one of which is for a single write request. The host computer comprises an I/O driver configured to generate individual requests to write data to memory locations within the I/O device in accordance with a software application executing in the host computer. Each write request generated by the I/O driver includes an address in the I/O device where data is to be written. A USB driver within the host computer is coupled to the USB and generates Data packets from the plurality of write requests posted in the queue. Once the Data packet is generated, the USB driver transmits the Data packet over the USB to the USB device. The host computer also includes a data structure stored in memory of the host computer for indicating postability of write requests posted within the queue. The data structure is accessible using addresses within the I/O device provided by the I/O driver write requests. A mini-driver coupled to the USB driver and data structure accesses the data structure using addresses of the write requests in order to determine the postability of the posted write request. If the mini-driver operating in connection with the data structure determines that the most recently posted write request is

directed to a memory location which is nonpostable, the mini-driver issues the flush command which causes the USB driver to transmit a Data packet generated from the posted writes. A mini-driver is also configured to monitor storage availability within the queue after each write is posted. If the mini-driver determines that the queue is incapable of receiving further write requests as a result of a lack of sufficient storage space, the mini-driver initiates the USB driver to transmit a Data packet generated from the posted write request.

Detailed Description Text (6):
USB device 30 includes: device interface 22 which, in one embodiment, is a CPU with a USB port; firmware 24, and; I/O device 14. I/O device 14 includes memory (not shown) accessible by device interface 22 acting in accordance with firmware 24 and commands received from host computer 12.

CLAIMS:

5. A universal serial bus system comprising:

a universal serial bus (USB) for transmitting data between devices coupled to the USB;

a USB device coupled to the USB, the USB device having an addressable memory for storing data;

a host computer coupled to the USB, wherein the host computer is configured to generate individual requests to write data to memory of the USB device;

a queue for posting host computer generated requests to write data to memory of the USB device;

wherein the host computer is configured to generate a data packet for transmission over the USB to the USB device, wherein the data packet is generated from the write requests posted in the queue, and wherein the

host computer is configured to generate the data packet when the queue lacks space to post further write requests.

6. The system of claim 5 further comprising a memory coupled to the host computer, for storing a data structure containing information indicating postability of host generated write requests in the queue, wherein the data structure is accessible using memory addresses of the USB where data is to be written.

7. The system of claim 6 wherein the host computer is configured to access the data structure using memory addresses in the USB device where data is to be written, wherein the host computer generates the data packet in response to the data structure indicating non-postability of a write request most recently posted in the queue.

12. In a system including a host computer and USB devices coupled to the host computer via a universal serial bus (USB), wherein said host computer generates requests to write data to the USB devices, a method comprising:

the host computer generating a first request to write a first data to memory in a USB device at a first memory address;

the host computer posting the first request in a queue;

the host computer generating a second request to write second data to memory in the USB device at a second memory address;

the host computer posting the second request in the queue;

the host computer determining whether the second data is postable;

the host computer generating a data packet from the first and second requests stored

in the queue in response to the host computer determining the second data to be nonpostable;

the host computer sending the data packet over the USB to the USB device, and;

the USB device writing the first and second data into memory of the USB device in response to the USB device receiving the data packet from the host computer.

15. The method of claim 12 further comprising:

the host computer creating a data structure including information indicating postability of data to be written to memory in the USB device, wherein postability information in the data structure is accessible using memory addresses in the USB device where data is to be written;

the host computer accessing the data structure with the second address to determine whether the second data is postable;

wherein the host computer generates the data packet in response to the data structure indicating the second data is nonpostable.

WEST

Generate Collection | Print

L2: Entry 22 of 26 File: USPT Jun 6, 2000

US-PAT-NO: 6073205
DOCUMENT-IDENTIFIER: US 6073205 A

TITLE: System and method of write posting in a universal serial bus system

DATE-ISSUED: June 6, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Thomson; Andrew	Austin	TX		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
National Instruments Corporation	Austin	TX			02

APPL-NO: 08/ 891150 [PALM]
DATE FILED: July 10, 1997

INT-CL: [07] G06 F 13/14

US-CL-ISSUED: 711/100; 711/168, 711/169, 711/147, 710/53, 710/33, 710/113
US-CL-CURRENT: 711/100; 710/113, 710/33, 710/53, 711/147, 711/168, 711/169

FIELD-OF-SEARCH: 395/101, 395/840-841, 395/872-877, 395/118, 711/149, 711/168, 711/169, 711/146, 711/100, 711/148, 710/3, 710/52, 710/113, 710/33

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4258418	March 1981	Heath	710/53
☐	4805094	February 1989	Oye et al.	710/53
☐	4860292	August 1989	Newman et al.	714/748
☐	5163132	November 1992	DuLac et al.	395/873
☐	5224213	June 1993	Dieffenderfer et al.	395/873
☐	5239644	August 1993	Seki et al.	710/52
☐	5394531	February 1995	Smith	711/136
☐	5487171	January 1996	Dodt et al.	710/20
☐	5539915	July 1996	Burton et al.	395/841
☐	5590310	December 1996	Willenz et al.	711/146
☐	5594926	January 1997	Chang et al.	710/52
☐	5604843	February 1997	Shaw et al.	395/101
☐	5619723	April 1997	Jones et al.	710/3
☐	5630166	May 1997	Gamache et al.	712/29
☐	5651137	July 1997	MacWilliams et al.	711/141
☐	5793992	August 1998	Steele et al.	710/113
☐	5832300	November 1998	Lowthert	710/33
☐	5845152	December 1998	Anderson et al.	710/52

OTHER PUBLICATIONS

Universal Serial Bus Specification, revision 0.9, Mar. 31, 1995, pp. 1-25.
Universal Serial Bus Bulletin, vol. 2, No. 3, Aug. 1996, pp. 1-8.

ART-UNIT: 279

PRIMARY-EXAMINER: Yoo; Do Hyun

ASSISTANT-EXAMINER: Nguyen; Than

ATTY-AGENT-FIRM: Conley, Rose & Tayon, P.C. Hood; Jeffrey C. Stephenson; Eric A.

ABSTRACT:

An apparatus and method for write posting in a universal serial bus (USB) system includes a host computer connected to USB devices via a USB. The host computer generates requests to write data to memory within the USB device. The host computer includes a queue for posting the write requests on generation thereof. The write requests are posted in the queue until the host computer transmits a single data packet generated from the posted write requests. The Data packet is generated in response to the host computer generating a request to read data from the USB device, the host computer determining that the most recently posted write request is directed to a memory location within the USB device which is nonpostable, or an indication that the queue lacks storage space for subsequent write requests. The USB device receives the transmitted Data packet from the host computer and writes data to internal memory locations in accordance with the received Data packet.

16 Claims, 5 Drawing figures

L2: Entry 23 of 26 File: USPT Jan 4, 2000

DOCUMENT-IDENTIFIER: US 6012103 A
TITLE: Bus interface system and method

Brief Summary Text (5):
When a peripheral device is first connected to the USB and the host computer through a standard USB communications port, the presence of the connected peripheral device is detected and a configuration process of the USB for the connected peripheral device, known as device enumeration, begins. The enumeration process assigns a unique USB address to the connected peripheral device, queries the connected peripheral device about its requirements and capabilities, writes data about the connected peripheral device into the host computer's operating system, and loads the appropriate software device driver from a storage location into the host computer's operating system. During the query, a data table stored in the peripheral device, which contains the particular peripheral device's configuration information, is read from the peripheral device into the host computer's memory. Upon completion of the enumeration process, the connected peripheral device is recognized by the host computer's operating system and may be used by application software being executed by the microprocessor of the host computer. The association of the device with the software device driver cannot be subsequently changed.

Brief Summary Text (6):
In a serial bus system, such as the USB, the only opportunity for associating software device drivers with a peripheral device is at the time when the peripheral device is plugged into the USB and the enumeration process occurs. Thus, to alter the configuration or personality of a peripheral device, such as downloading new code or configuration information into the memory of the peripheral device, the host computer system must detect a peripheral device connection or a disconnection and then a reconnection.

Brief Summary Text (12):
In addition, the universal serial bus interface system and method may be a single semiconductor chip which may be incorporated into a plurality of peripheral devices made by a plurality of manufacturers. The chip may initially have a generic configuration (e.g., not specific to a particular peripheral device). Then, the appropriate configuration information for a particular peripheral device and manufacturer may be downloaded to the chip, an electronic simulation of the disconnection and reconnection of the peripheral device occurs, the peripheral device is recognized as a new, manufacturer specific peripheral device and the appropriate software device driver is loaded into the memory of the host computer.

Detailed Description Text (3):
FIG. 1 is a diagram illustrating a standardized bus interface, such as a conventional computer system 20, that may include a host computer system 22 and a peripheral device 24. The peripheral device is connected to the host computer by a universal serial bus (USB) 26. The host computer may include a central processing unit (CPU) 28 connected to a USB interface (I/F) circuit 30, and the USB standard provides a universal electrical and physical interface for the peripheral devices via bus 26. The CPU executes software application code located in a memory 31 and communicates data to and from the peripheral device through the USB interface and the USB 26. The host computer may also include an operating system 32 which may include a software device driver 33. The peripheral device 24 may include a USB interface circuit 34, a CPU 36 and a non-volatile memory 38 that may store

configuration information describing the characteristics of the peripheral device. The non-volatile memory may be a read only memory (ROM) or an erasable programmable read only memory (EPROM).

Detailed Description Text (4):
When the peripheral device is initially connected to the USB, an enumeration process is conducted in which the host computer determines the characteristics of the peripheral device by receiving the configuration information from the memory 38 within the peripheral device, and configures the USB according to the characteristics of the peripheral device. As shown, the configuration information about the characteristics of the peripheral device in a conventional USB system is stored in a non-volatile memory 38 on the peripheral device. The data about the characteristics of the peripheral device is programmed into the non-volatile memory at the factory, and the characteristics of the peripheral device may not be easily altered. In addition, the memory in the peripheral device stores all of the configuration information about the peripheral device which may require a large amount of memory in the peripheral device.

Detailed Description Text (6):
FIG. 2 is a diagram illustrating a computer system 50 that may have a universal serial bus system in accordance with the invention. The computer system may include a host computer 52 connected to a peripheral device 54 by a universal serial bus (USB) 60. The host computer may include a CPU 62, a memory 64, an operating system 65 and a USB interface circuit 66. One or more peripheral device drivers, such as a first peripheral device driver 68, may be stored in the operating system 65. Each device driver contains information about the proper configuration of the USB for a particular class of peripheral devices. The operating system within the host computer may also contain a plurality of different configuration information sets 70, which may include configuration data for a particular peripheral device (including which device driver to use), microprocessor code to be executed by a CPU located in the peripheral device, or logic configuration data to configure logic circuits in the peripheral device. This invention advantageously enables these configuration information sets to be updated or altered easily since they are located in the host computer and not in a non-volatile memory in the peripheral device.

Detailed Description Text (9):
In operation, during the initial factory configuration of the peripheral device with the USB interface system in accordance with the invention, the memory may store an identification code indicating the appropriate configuration information set to be loaded. Thus, when the peripheral device is first connected to the USB, the configuration information 70, including any microprocessor code applicable to the peripheral device and the appropriate configuration data for the peripheral device may be downloaded over the USB into the memory 74 of the peripheral device 54 as shown by the dashed arrow 78. The electrical simulation of the disconnection and reconnection of the peripheral device from the USB, as described below, may be initiated and a re-enumeration process may occur. During the re-enumeration process, the newly downloaded configuration information may be used to reconfigure the USB for the peripheral device and the host computer may select the appropriate software device driver 68 for the peripheral device based on the configuration information and load the device driver into memory 64 as shown by arrow 80. For example, a plurality of different peripheral devices manufactured by different companies may each include a USB interface system in accordance with the invention. The USB interface system for each peripheral device is identical (e.g. has a USB interface circuit and a memory) except that each memory may contain an identification code that is unique to, for example, a particular manufacturer. When one of the peripheral devices is connected to the USB and the host computer, the appropriate configuration information for the peripheral device, based on the identification code, is downloaded over the USB to the memory of the peripheral device and the appropriate software device driver is loaded into the memory of the host computer. Thus, a plurality of different peripheral device may include the same USB interface system hardware since the configuration information is located in the operating system of the host computer. Now, a conventional USB interface circuit and the universal serial bus interface circuit in accordance with the invention that permits the electronic disconnection and reconnection of the peripheral device will be

described.

Detailed Description Text (14):
FIGS. 5, 6, and 7 are diagrams illustrating three different peripheral devices from different manufacturers, for example, being connected to a host computer in accordance with the invention. In each Figure, a computer system 140 may include a host computer 142, a plurality of peripheral devices, such as peripheral device "A" 144 (shown in FIG. 5), peripheral device "B" 146 (shown in FIG. 6), peripheral device "C" 148 (shown in FIG. 7) and a USB bus 149. The host computer 142 may include a CPU 150, a memory 152, an operating system 154 and a USB interface circuit 156. The operating system, in this example, may include a plurality of software device drivers, such as device driver "A" 158, device driver "B" 160 and device driver "C" 162, and a plurality of configuration information sets, such as device "A" characteristics 164, device "B" characteristics 166 and device "C" characteristics 168.

Detailed Description Text (16):
As shown in FIG. 5, peripheral device "A" 144 may have a unique manufacturer signature in the non-volatile memory 178. When the peripheral device is connected to the computer system, the enumeration process begins in which the USB interface system 170 is recognized by the USB as a generic device and the unique manufacturer signature is read from the non-volatile memory by the CPU 150 over the USB 149. The unique signature identifies device "A" characteristics 164 as the appropriate configuration information and that configuration information may be downloaded over the USB 149 into the memory 174 of the peripheral device as shown by dashed arrow 180. Then the electrical simulation of the disconnection and reconnection of the peripheral device occurs, as described above, which cause re-enumeration of the peripheral device. During re-enumeration, device driver "A" 158, which is identified by device "A" characteristics 164 as the appropriate device driver, is loaded from the operating system into the memory, as shown by arrow 182, such that the peripheral device is now recognized as a peripheral device with device "A" characteristics. Thus, a generic hardware USB interface system may be incorporated into a peripheral device and the particular characteristics for the particular peripheral device may be later downloaded from the host computer into the peripheral device.

Detailed Description Text (17):
Similarly, as shown in FIGS. 6 and 7, the peripheral devices 146, 148 may include the generic USB interface system and a unique manufacturer signature in the non-volatile memory and may be re-enumerated in accordance with the invention so that the appropriate device characteristics are downloaded from the host computer over the USB into the memory of the peripheral device and the appropriate device driver may be selected by the host computer. Thus, peripheral device "B" 146 (shown in FIG. 6) may have device "B" characteristics 166 downloaded into its memory, as shown by dashed arrow 184 in FIG. 6, and the host computer may use device driver "B" 160, as shown by arrow 186 in FIG. 6. The peripheral device "C" 148 (shown in FIG. 7) may have device "C" characteristics 168 downloaded into its memory as shown by the dashed arrow 188 in FIG. 7 and the host computer may use device driver "C" 162 as shown by arrow 190 in FIG. 7. Thus, a generic USB interface system may be incorporated into a plurality of different peripheral devices, the appropriate configuration information may be downloaded into the peripheral device, and the re-enumeration recognizes the peripheral device as a manufacturer specific device. The re-enumeration of the generic peripheral device ensures that the host computer discards all device driver information about the generic connection and loads the manufacturer-specific device driver software.

Detailed Description Text (19):
The universal serial bus interface system in accordance with the invention provides several advantages. The system provides an easy technique to associate new device driver software with a peripheral device, known as re-enumeration. A peripheral device may be disconnected and reconnected to the USB without the physical disconnection and reconnection of the peripheral device to cause re-enumeration to occur. In addition, because the peripheral device is not physically disconnected from the host computer, the peripheral device may use the electrical power available over the USB bus to maintain the configuration information if it is in a volatile

memory and to perform tasks during the simulated disconnection. The characteristics of the peripheral devices contained in one or more configuration information sets may be stored in the host computer so that the configuration information may be easily changed. The combination of the configuration information sets stored in the host computer and the electronic disconnection and reconnection of the peripheral device permits the characteristics of a peripheral device to be changed rapidly without physical disconnection of the peripheral device. In addition, the configuration of any peripheral device connected to the USB may be altered or changed multiple times. The system also permits a generic USB interface system to be incorporated into a plurality of peripheral devices and then each peripheral device to be configured with manufacturer specific configuration information.

WEST

Generate Collection | Print

L2: Entry 23 of 26 File: USPT Jan 4, 2000

US-PAT-NO: 6012103
DOCUMENT-IDENTIFIER: US 6012103 A

TITLE: Bus interface system and method

DATE-ISSUED: January 4, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Sartore; Ronald H.	San Diego	CA		
Larky; Steven P.	Del Mar	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Cypress Semiconductor Corp.	San Jose	CA			02

APPL-NO: 08/ 886923 [PALM]
DATE FILED: July 2, 1997

INT-CL: [06] G06 F 13/368

US-CL-ISSUED: 710/8; 710/9, 710/10
US-CL-CURRENT: 710/8; 710/10, 710/9

FIELD-OF-SEARCH: 379/395, 709/220, 709/250, 711/114, 711/115, 711/130, 713/100, 710/9

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4641261	February 1987	Dwyer et al.	
☐	4862355	August 1989	Newman et al.	
☐	5289580	February 1994	Latif et al.	
☐	5428748	June 1995	Davidson et al.	710/9
☐	5488657	January 1996	Lynn et al.	379/395
☐	5574859	November 1996	Yeh	
☐	5577213	November 1996	Avery et al.	
☐	5586268	December 1996	Chen et al.	
☐	5606672	February 1997	Wade	
☐	5615344	March 1997	Corder	
☐	5634074	May 1997	Devon et al.	395/828
☐	5671355	September 1997	Collins	709/250
☐	5687346	November 1997	Shinohara	711/130
☐	5701429	December 1997	Legvold et al.	711/114
☐	5794033	August 1998	Aldebert et al.	713/100
☐	5802558	September 1998	Pierce	711/115
☐	5838907	November 1998	Hansen	709/220

ART-UNIT: 272

PRIMARY-EXAMINER: Lee; Thomas C.

ASSISTANT-EXAMINER: Yuan; Chien

ATTY-AGENT-FIRM: Maiorana, P.C.; Christopher P.

ABSTRACT:

A system and method for reconfiguring a peripheral device connected by a computer bus and port to a host from a first generic configuration to a second manufacturer specific configuration is provided in which the configuration of a peripheral device may be electronically reset. A peripheral interface device for a standardized computer peripheral device bus and port is also provided in which a physical disconnection and reconnection of the peripheral device is emulated to reconfigure the bus and port for a particular peripheral device.

33 Claims, 8 Drawing figures

First Hit Fwd Refs
End of Result Set

☐ Generate Collection Print

L4: Entry 1 of 1 File: USPT Jan 4, 2000

DOCUMENT-IDENTIFIER: US 6012103 A
TITLE: Bus interface system and method

Detailed Description Text (3):
FIG. 1 is a diagram illustrating a standardized bus interface, such as a conventional computer system 20, that may include a host computer system 22 and a peripheral device 24. The peripheral device is connected to the host computer by a universal serial bus (USB) 26. The host computer may include a central processing unit (CPU) 28 connected to a USB interface (I/F) circuit 30, and the USB standard provides a universal electrical and physical interface for the peripheral devices via bus 26. The CPU executes software application code located in a memory 31 and communicates data to and from the peripheral device through the USB interface and the USB 26. The host computer may also include an operating system 32 which may include a software device driver 33. The peripheral device 24 may include a USB interface circuit 34, a CPU 36 and a non-volatile memory 38 that may store configuration information describing the characteristics of the peripheral device. The non-volatile memory may be a read only memory (ROM) or an erasable programmable read only memory (EPROM).

Detailed Description Text (4):
When the peripheral device is initially connected to the USB, an enumeration process is conducted in which the host computer determines the characteristics of the peripheral device by receiving the configuration information from the memory 38 within the peripheral device, and configures the USB according to the characteristics of the peripheral device. As shown, the configuration information about the characteristics of the peripheral device in a conventional USB system is stored in a non-volatile memory 38 on the peripheral device. The data about the characteristics of the peripheral device is programmed into the non-volatile memory at the factory, and the characteristics of the peripheral device may not be easily altered. In addition, the memory in the peripheral device stores all of the configuration information about the peripheral device which may require a large amount of memory in the peripheral device.

Detailed Description Text (6):
FIG. 2 is a diagram illustrating a computer system 50 that may have a universal serial bus system in accordance with the invention. The computer system may include a host computer 52 connected to a peripheral device 54 by a universal serial bus (USB) 60. The host computer may include a CPU 62, a memory 64, an operating system 65 and a USB interface circuit 66. One or more peripheral device drivers, such as a first peripheral device driver 68, may be stored in the operating system 65. Each device driver contains information about the proper configuration of the USB for a particular class of peripheral devices. The operating system within the host computer may also contain a plurality of different configuration information sets 70, which may include configuration data for a particular peripheral device (including which device driver to use), microprocessor code to be executed by a CPU located in the peripheral device, or logic configuration data to configure logic circuits in the peripheral device. This invention advantageously enables these configuration information sets to be updated or altered easily since they are

located in the host computer and not in a non-volatile memory in the peripheral device.

Detailed Description Text (15):
Each peripheral device 144, 146, 148 may include a universal USB interface system 170 that may include a USB interface circuit 172 as shown in FIG. 4 and a loadable memory 174. Each peripheral device may also include a CPU 176 and a non-volatile memory 178. The non-volatile memory may store a unique manufacture signature or identifier that identifies the appropriate configuration information to be downloaded into the peripheral device. Now, the connection of each of these peripheral device to the computer system will be described.

Detailed Description Text (16):
As shown in FIG. 5, peripheral device "A" 144 may have a unique manufacturer signature in the non-volatile memory 178. When the peripheral device is connected to the computer system, the enumeration process begins in which the USB interface system 170 is recognized by the USB as a generic device and the unique manufacturer signature is read from the non-volatile memory by the CPU 150 over the USB 149. The unique signature identifies device "A" characteristics 164 as the appropriate configuration information and that configuration information may be downloaded over the USB 149 into the memory 174 of the peripheral device as shown by dashed arrow 180. Then the electrical simulation of the disconnection and reconnection of the peripheral device occurs, as described above, which cause re-enumeration of the peripheral device. During re-enumeration, device driver "A" 158, which is identified by device "A" characteristics 164 as the appropriate device driver, is loaded from the operating system into the memory, as shown by arrow 182, such that the peripheral device is now recognized as a peripheral device with device "A" characteristics. Thus, a generic hardware USB interface system may be incorporated into a peripheral device and the particular characteristics for the particular peripheral device may be later downloaded from the host computer into the peripheral device.

Detailed Description Text (17):
Similarly, as shown in FIGS. 6 and 7, the peripheral devices 146, 148 may include the generic USB interface system and a unique manufacturer signature in the non-volatile memory and may be re-enumerated in accordance with the invention so that the appropriate device characteristics are downloaded from the host computer over the USB into the memory of the peripheral device and the appropriate device driver may be selected by the host computer. Thus, peripheral device "B" 146 (shown in FIG. 6) may have device "B" characteristics 166 downloaded into its memory, as shown by dashed arrow 184 in FIG. 6, and the host computer may use device driver "B" 160, as shown by arrow 186 in FIG. 6. The peripheral device "C" 148 (shown in FIG. 7) may have device "C" characteristics 168 downloaded into its memory as shown by the dashed arrow 188 in FIG. 7 and the host computer may use device driver "C" 162 as shown by arrow 190 in FIG. 7. Thus, a generic USB interface system may be incorporated into a plurality of different peripheral devices, the appropriate configuration information may be downloaded into the peripheral device, and the re-enumeration recognizes the peripheral device as a manufacturer specific device. The re-enumeration of the generic peripheral device ensures that the host computer discards all device driver information about the generic connection and loads the manufacturer-specific device driver software.

First Hit Fwd Refs

End of Result Set

Generate Collection Print

L5: Entry 2 of 2 File: USPT Jan 4, 2000

DOCUMENT-IDENTIFIER: US 6012103 A
TITLE: Bus interface system and method

Abstract Text (1):
A system and method for reconfiguring a peripheral device connected by a computer bus and port to a host from a first generic configuration to a second manufacturer specific configuration is provided in which the configuration of a peripheral device may be electronically reset. A peripheral interface device for a standardized computer peripheral device bus and port is also provided in which a physical disconnection and reconnection of the peripheral device is emulated to reconfigure the bus and port for a particular peripheral device.

Brief Summary Text (12):
In addition, the universal serial bus interface system and method may be a single semiconductor chip which may be incorporated into a plurality of peripheral devices made by a plurality of manufacturers. The chip may initially have a generic configuration (e.g., not specific to a particular peripheral device). Then, the appropriate configuration information for a particular peripheral device and manufacturer may be downloaded to the chip, an electronic simulation of the disconnection and reconnection of the peripheral device occurs, the peripheral device is recognized as a new, manufacturer specific peripheral device and the appropriate software device driver is loaded into the memory of the host computer.

Detailed Description Text (9):
In operation, during the initial factory configuration of the peripheral device with the USB interface system in accordance with the invention, the memory may store an identification code indicating the appropriate configuration information set to be loaded. Thus, when the peripheral device is first connected to the USB, the configuration information 70, including any microprocessor code applicable to the peripheral device and the appropriate configuration data for the peripheral device may be downloaded over the USB into the memory 74 of the peripheral device 54 as shown by the dashed arrow 78. The electrical simulation of the disconnection and reconnection of the peripheral device from the USB, as described below, may be initiated and a re-enumeration process may occur. During the re-enumeration process, the newly downloaded configuration information may be used to reconfigure the USB for the peripheral device and the host computer may select the appropriate software device driver 68 for the peripheral device based on the configuration information and load the device driver into memory 64 as shown by arrow 80. For example, a plurality of different peripheral devices manufactured by different companies may each include a USB interface system in accordance with the invention. The USB interface system for each peripheral device is identical (e.g. has a USB interface circuit and a memory) except that each memory may contain an identification code that is unique to, for example, a particular manufacturer. When one of the peripheral devices is connected to the USB and the host computer, the appropriate configuration information for the peripheral device, based on the identification code, is downloaded over the USB to the memory of the peripheral device and the appropriate software device driver is loaded into the memory of the host computer. Thus, a plurality of different peripheral device may include the

same USB interface system hardware since the configuration information is located in the operating system of the host computer. Now, a conventional USB interface circuit and the universal serial bus interface circuit in accordance with the invention that permits the electronic disconnection and reconnection of the peripheral device will be described.

Detailed Description Text (14):
FIGS. 5, 6, and 7 are diagrams illustrating three different peripheral devices from different manufacturers, for example, being connected to a host computer in accordance with the invention. In each Figure, a computer system 140 may include a host computer 142, a plurality of peripheral devices, such as peripheral device "A" 144 (shown in FIG. 5), peripheral device "B" 146 (shown in FIG. 6), peripheral device "C" 148 (shown in FIG. 7) and a USB bus 149. The host computer 142 may include a CPU 150, a memory 152, an operating system 154 and a USB interface circuit 156. The operating system, in this example, may include a plurality of software device drivers, such as device driver "A" 158, device driver "B" 160 and device driver "C" 162, and a plurality of configuration information sets, such as device "A" characteristics 164, device "B" characteristics 166 and device "C" characteristics 168.

Detailed Description Text (16):
As shown in FIG. 5, peripheral device "A" 144 may have a unique manufacturer signature in the non-volatile memory 178. When the peripheral device is connected to the computer system, the enumeration process begins in which the USB interface system 170 is recognized by the USB as a generic device and the unique manufacturer signature is read from the non-volatile memory by the CPU 150 over the USB 149. The unique signature identifies device "A" characteristics 164 as the appropriate configuration information and that configuration information may be downloaded over the USB 149 into the memory 174 of the peripheral device as shown by dashed arrow 180. Then the electrical simulation of the disconnection and reconnection of the peripheral device occurs, as described above, which cause re-enumeration of the peripheral device. During re-enumeration, device driver "A" 158, which is identified by device "A" characteristics 164 as the appropriate device driver, is loaded from the operating system into the memory, as shown by arrow 182, such that the peripheral device is now recognized as a peripheral device with device "A" characteristics. Thus, a generic hardware USB interface system may be incorporated into a peripheral device and the particular characteristics for the particular peripheral device may be later downloaded from the host computer into the peripheral device.

Detailed Description Text (17):
Similarly, as shown in FIGS. 6 and 7, the peripheral devices 146, 148 may include the generic USB interface system and a unique manufacturer signature in the non-volatile memory and may be re-enumerated in accordance with the invention so that the appropriate device characteristics are downloaded from the host computer over the USB into the memory of the peripheral device and the appropriate device driver may be selected by the host computer. Thus, peripheral device "B" 146 (shown in FIG. 6) may have device "B" characteristics 166 downloaded into its memory, as shown by dashed arrow 184 in FIG. 6, and the host computer may use device driver "B" 160, as shown by arrow 186 in FIG. 6. The peripheral device "C" 148 (shown in FIG. 7) may have device "C" characteristics 168 downloaded into its memory as shown by the dashed arrow 188 in FIG. 7 and the host computer may use device driver "C" 162 as shown by arrow 190 in FIG. 7. Thus, a generic USB interface system may be incorporated into a plurality of different peripheral devices, the appropriate configuration information may be downloaded into the peripheral device, and the re-enumeration recognizes the peripheral device as a manufacturer specific device. The re-enumeration of the generic peripheral device ensures that the host computer discards all device driver information about the generic connection and loads the manufacturer-specific device driver software.

Detailed Description Text (19):
The universal serial bus interface system in accordance with the invention provides several advantages. The system provides an easy technique to associate new device driver software with a peripheral device, known as re-enumeration. A peripheral device may be disconnected and reconnected to the USB without the physical disconnection and reconnection of the peripheral device to cause re-enumeration to occur. In addition, because the peripheral device is not physically disconnected from the host computer, the peripheral device may use the electrical power available over the USB bus to maintain the configuration information if it is in a volatile memory and to perform tasks during the simulated disconnection. The characteristics of the peripheral devices contained in one or more configuration information sets may be stored in the host computer so that the configuration information may be easily changed. The combination of the configuration information sets stored in the host computer and the electronic disconnection and reconnection of the peripheral device permits the characteristics of a peripheral device to be changed rapidly without physical disconnection of the peripheral device. In addition, the configuration of any peripheral device connected to the USB may be altered or changed multiple times. The system also permits a generic USB interface system to be incorporated into a plurality of peripheral devices and then each peripheral device to be configured with manufacturer specific configuration information.

CLAIMS:

2. The system of claim 1, wherein said first configuration is a generic configuration assigned to the peripheral device and said second configuration comprises a unique manufacturer configuration.

15. The method of claim 14, wherein said first configuration comprises generic configuration assigned to the peripheral device and said second configuration comprises a unique manufacturer configuration.

30. The device of claim 29, wherein said first configuration comprises a generic configuration and said second configuration comprises a manufacturer specific configuration.

First Hit Fwd Refs

End of Result Set

☐ Generate Collection Print

L3: Entry 1 of 1 File: USPT Jan 4, 2000

DOCUMENT-IDENTIFIER: US 6012103 A
TITLE: Bus interface system and method

Abstract Text (1):
A system and method for reconfiguring a peripheral device connected by a computer bus and port to a host from a first generic configuration to a second manufacturer specific configuration is provided in which the configuration of a peripheral device may be electronically reset. A peripheral interface device for a standardized computer peripheral device bus and port is also provided in which a physical disconnection and reconnection of the peripheral device is emulated to reconfigure the bus and port for a particular peripheral device.

Brief Summary Text (13):
In accordance with the invention, a system and method for reconfiguring a peripheral device connected by a computer bus and port to a host computer is provided in which a host computer detects a peripheral device connected to the port in which the peripheral device has a first configuration. A second set of configuration information may be downloaded from the host computer into the peripheral device over the computer bus, and the configuration of the peripheral device is electronically reset from said first configuration to a second configuration based on the second set of configuration information.

CLAIMS:

13. The system of claim 1, wherein said third circuit comprises a reset circuit configured to reset the configuration of the peripheral device.

23. The method of claim 14, wherein said simulating comprises electronically resetting the configuration of the peripheral device, controllable by the peripheral device.

33. The device of claim 24, wherein said electronic simulating means comprises means for electronically resetting the configuration of the peripheral device.

First Hit Fwd Refs

End of Result Set

☐ Generate Collection Print

L1: Entry 1 of 1 File: USPT Jan 4, 2000

DOCUMENT-IDENTIFIER: US 6012103 A
TITLE: Bus interface system and method

Abstract Text (1):
A system and method for reconfiguring a peripheral device connected by a computer bus and port to a host from a first generic configuration to a second manufacturer specific configuration is provided in which the configuration of a peripheral device may be electronically reset. A peripheral interface device for a standardized computer peripheral device bus and port is also provided in which a physical disconnection and reconnection of the peripheral device is emulated to reconfigure the bus and port for a particular peripheral device.

Brief Summary Text (12):
In addition, the universal serial bus interface system and method may be a single semiconductor chip which may be incorporated into a plurality of peripheral devices made by a plurality of manufacturers. The chip may initially have a generic configuration (e.g., not specific to a particular peripheral device). Then, the appropriate configuration information for a particular peripheral device and manufacturer may be downloaded to the chip, an electronic simulation of the disconnection and reconnection of the peripheral device occurs, the peripheral device is recognized as a new, manufacturer specific peripheral device and the appropriate software device driver is loaded into the memory of the host computer.

Detailed Description Text (9):
In operation, during the initial factory configuration of the peripheral device with the USB interface system in accordance with the invention, the memory may store an identification code indicating the appropriate configuration information set to be loaded. Thus, when the peripheral device is first connected to the USB, the configuration information 70, including any microprocessor code applicable to the peripheral device and the appropriate configuration data for the peripheral device may be downloaded over the USB into the memory 74 of the peripheral device 54 as shown by the dashed arrow 78. The electrical simulation of the disconnection and reconnection of the peripheral device from the USB, as described below, may be initiated and a re-enumeration process may occur. During the re-enumeration process, the newly downloaded configuration information may be used to reconfigure the USB for the peripheral device and the host computer may select the appropriate software device driver 68 for the peripheral device based on the configuration information and load the device driver into memory 64 as shown by arrow 80. For example, a plurality of different peripheral devices manufactured by different companies may each include a USB interface system in accordance with the invention. The USB interface system for each peripheral device is identical (e.g. has a USB interface circuit and a memory) except that each memory may contain an identification code that is unique to, for example, a particular manufacturer. When one of the peripheral devices is connected to the USB and the host computer, the appropriate configuration information for the peripheral device, based on the identification code, is downloaded over the USB to the memory of the peripheral device and the appropriate software device driver is loaded into the memory of the host computer. Thus, a plurality of different peripheral device may include the

same USB interface system hardware since the configuration information is located in the operating system of the host computer. Now, a conventional USB interface circuit and the universal serial bus interface circuit in accordance with the invention that permits the electronic disconnection and reconnection of the peripheral device will be described.

Detailed Description Text (14):
FIGS. 5, 6, and 7 are diagrams illustrating three different peripheral devices from different manufacturers, for example, being connected to a host computer in accordance with the invention. In each Figure, a computer system 140 may include a host computer 142, a plurality of peripheral devices, such as peripheral device "A" 144 (shown in FIG. 5), peripheral device "B" 146 (shown in FIG. 6), peripheral device "C" 148 (shown in FIG. 7) and a USB bus 149. The host computer 142 may include a CPU 150, a memory 152, an operating system 154 and a USB interface circuit 156. The operating system, in this example, may include a plurality of software device drivers, such as device driver "A" 158, device driver "B" 160 and device driver "C" 162, and a plurality of configuration information sets, such as device "A" characteristics 164, device "B" characteristics 166 and device "C" characteristics 168.

Detailed Description Text (16):
As shown in FIG. 5, peripheral device "A" 144 may have a unique manufacturer signature in the non-volatile memory 178. When the peripheral device is connected to the computer system, the enumeration process begins in which the USB interface system 170 is recognized by the USB as a generic device and the unique manufacturer signature is read from the non-volatile memory by the CPU 150 over the USB 149. The unique signature identifies device "A" characteristics 164 as the appropriate configuration information and that configuration information may be downloaded over the USB 149 into the memory 174 of the peripheral device as shown by dashed arrow 180. Then the electrical simulation of the disconnection and reconnection of the peripheral device occurs, as described above, which cause re-enumeration of the peripheral device. During re-enumeration, device driver "A" 158, which is identified by device "A" characteristics 164 as the appropriate device driver, is loaded from the operating system into the memory, as shown by arrow 182, such that the peripheral device is now recognized as a peripheral device with device "A" characteristics. Thus, a generic hardware USB interface system may be incorporated into a peripheral device and the particular characteristics for the particular peripheral device may be later downloaded from the host computer into the peripheral device.

Detailed Description Text (17):
Similarly, as shown in FIGS. 6 and 7, the peripheral devices 146, 148 may include the generic USB interface system and a unique manufacturer signature in the non-volatile memory and may be re-enumerated in accordance with the invention so that the appropriate device characteristics are downloaded from the host computer over the USB into the memory of the peripheral device and the appropriate device driver may be selected by the host computer. Thus, peripheral device "B" 146 (shown in FIG. 6) may have device "B" characteristics 166 downloaded into its memory, as shown by dashed arrow 184 in FIG. 6, and the host computer may use device driver "B" 160, as shown by arrow 186 in FIG. 6. The peripheral device "C" 148 (shown in FIG. 7) may have device "C" characteristics 168 downloaded into its memory as shown by the dashed arrow 188 in FIG. 7 and the host computer may use device driver "C" 162 as shown by arrow 190 in FIG. 7. Thus, a generic USB interface system may be incorporated into a plurality of different peripheral devices, the appropriate configuration information may be downloaded into the peripheral device, and the re-enumeration recognizes the peripheral device as a manufacturer specific device. The re-enumeration of the generic peripheral device ensures that the host computer discards all device driver information about the generic connection and loads the manufacturer-specific device driver software.

Detailed Description Text (19):
The universal serial bus interface system in accordance with the invention provides several advantages. The system provides an easy technique to associate new device driver software with a peripheral device, known as re-enumeration. A peripheral device may be disconnected and reconnected to the USB without the physical disconnection and reconnection of the peripheral device to cause re-enumeration to occur. In addition, because the peripheral device is not physically disconnected from the host computer, the peripheral device may use the electrical power available over the USB bus to maintain the configuration information if it is in a volatile memory and to perform tasks during the simulated disconnection. The characteristics of the peripheral devices contained in one or more configuration information sets may be stored in the host computer so that the configuration information may be easily changed. The combination of the configuration information sets stored in the host computer and the electronic disconnection and reconnection of the peripheral device permits the characteristics of a peripheral device to be changed rapidly without physical disconnection of the peripheral device. In addition, the configuration of any peripheral device connected to the USB may be altered or changed multiple times. The system also permits a generic USB interface system to be incorporated into a plurality of peripheral devices and then each peripheral device to be configured with manufacturer specific configuration information.

CLAIMS:

2. The system of claim 1, wherein said first configuration is a generic configuration assigned to the peripheral device and said second configuration comprises a unique manufacturer configuration.

15. The method of claim 14, wherein said first configuration comprises generic configuration assigned to the peripheral device and said second configuration comprises a unique manufacturer configuration.

30. The device of claim 29, wherein said first configuration comprises a generic configuration and said second configuration comprises a manufacturer specific configuration.

L2: Entry 14 of 26 File: USPT Apr 17, 2001

DOCUMENT-IDENTIFIER: US 6219736 B1
TITLE: Universal serial bus (USB) RAM architecture for use with microcomputers via an interface optimized for integrated services device network (ISDN)

Brief Summary Text (7):
The Universal Serial Bus (USB) and "firewire" (IEEE 1394) has been introduced in the computer industry to effectuate "time sharing" of many of these low speed peripheral devices over a single higher speed connection thereby providing higher performance communication links while using such peripheral devices. This higher speed connection requires only minimal resources (such as I/O, DMA, Interrupt and Memory) from the host system. Prior art systems require such resources per peripheral.

Detailed Description Text (24):
Handshake responses to USB packets must occur in approximately one microsecond. Even with interrupts, this is considerably faster than typical microcomputers can respond, therefore the architecture must compensate for this mismatch. The preferred compensation implementation employ the use of `auto-NAK` wherever feasible, and the use of "pre-configuration." For example, only one pipe at a time can be active, therefore, in principle one working pointer will suffice to read and write packets in the USB RAM 130. However, if the microcomputer cannot respond fast enough to setup the pointer for each pipe, then the pointer would have to point to the same default memory location for all pipes. But this then requires the microcomputer to load and unload each data packet quickly and to free up the memory for the next transaction. In general, this is not practical, therefore we use the "pre-configure" strategy: each endpoint has an associated virtual endpoint register, within the virtual endpoint register file storage location 248 of the dual port RAM device 214, identified within the device 214 by a pre-assigned address per each endpoint register. Using this address, each endpoint register may be pre-loaded during reset by the microcontroller device 140.

WEST

Generate Collection | Print

L2: Entry 14 of 26 File: USPT Apr 17, 2001

US-PAT-NO: 6219736
DOCUMENT-IDENTIFIER: US 6219736 B1

TITLE: Universal serial bus (USB) RAM architecture for use with microcomputers via an interface optimized for integrated services device network (ISDN)

DATE-ISSUED: April 17, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Klingman; Edwin E.	San Gregorio	CA	94074	

APPL-NO: 09/ 191443 [PALM]
DATE FILED: November 12, 1998

PARENT-CASE:
CROSS REFERENCE TO RELATED APPLICATION This application is a continuation-in-part of my prior application Ser. No. 08/846,118, filed Apr. 24, 1997, now U.S. Pat. No. 5,860,021, entitled "A SINGLE CHIP MICROCONTROLLER HAVING DOWN-LOADABLE MEMORY ORGANIZATION SUPPORTING "SHADOW" PERSONALITY, OPTIMIZED FOR BI-DIRECTIONAL DATA TRANSFERS OVER A COMMUNICATION CHANNEL."

INT-CL: [07] G06 F 13/00

US-CL-ISSUED: 710/129; 710/14, 710/52, 710/100, 710/128, 709/226, 709/250, 370/259, 370/420, 370/524
US-CL-CURRENT: 710/315; 370/259, 370/420, 370/524, 709/226, 709/250, 710/100, 710/14, 710/52

FIELD-OF-SEARCH: 710/52, 710/14, 710/100, 710/128, 710/129, 709/250, 709/226, 370/259, 370/420, 370/524

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4604683	August 1986	Russ et al.	710/100
☐	4799156	January 1989	Shavit et al.	705/26
☐	5309440	May 1994	Nakamura et al.	370/420
☐	5497373	March 1996	Hulen et al.	370/259
☐	5530894	June 1996	Farrell et al.	709/250
☐	5537654	July 1996	Bedingfield et al.	710/14
☐	5541930	July 1996	Klingman	370/524
☐	5574861	November 1996	Lorvig et al.	709/226
☐	5682552	October 1997	Kuboki et al.	710/52
☐	5859993	January 1999	Snyder	712/208
☐	5974486	October 1999	Siddappa	710/53
☐	6044428	March 2000	Rayabhari	710/129

OTHER PUBLICATIONS

Don Johnson, "Universal Serial Bus System Architecture".

ART-UNIT: 272

PRIMARY-EXAMINER: Lee; Thomas C.

ASSISTANT-EXAMINER: Peyton; Tammara

ATTY-AGENT-FIRM: Oppenheimer, Wolff & Donnelly, LLP Hamrick; Claude A. S.

ABSTRACT:

A RAM-based interrupt-driven interface device is disclosed for establishing a communication link between a universal serial bus (USB) host and a microcontroller device for providing a control function, the interface device being operative to receive digital information in the form of command, data and control packets from the host and to process the packets and communicate the processed digital information to the microcontroller device, and in response thereto, the microcontroller device being operative to communicate digital information to the interface device for processing and transfer thereof to the host. The interface device includes means for receiving a command generated by the host through a USB bus, means for storing the host-generated command and for generating an interface device interrupt signal upon storage of said host-generated command for use by the microcontroller device in responding to the host-generated command, a microcontroller bus for transferring microcontroller information and the interface device interrupt signal between the interface device and the microcontroller device. The interface device further includes means for receiving a microcontroller command from the microcontroller device in response to said interface device interrupt signal and means for storing the microcontroller command and it is operative to generate a microcontroller device interrupt signal upon storage of the microcontroller command for use by the interface device in developing an address for identification of the interface device to the host during subsequent communications therebetween, wherein during communication between the host and the interface device, the interface device-developed address is used by the interface device to identify host-provided information in the form of packets, and upon processing of the host-provided information, to provide the microcontroller device with the necessary information to allow it to respond to the host thereby allowing a generic microcontroller device to be flexibly interfaced with a USB, host for communication therebetween.

First Hit Fwd Refs
End of Result Set

☐ Generate Collection Print

L2: Entry 1 of 1 File: USPT Apr 17, 2001

DOCUMENT-IDENTIFIER: US 6219736 B1
TITLE: Universal serial bus (USB) RAM architecture for use with microcomputers via an interface optimized for integrated services device network (ISDN)

Brief Summary Text (46):
Accordingly, it is an objection of the present invention to provide a Serial Interface Controller that uses buffering via a memory-based interface capable of generating interrupt signals to the generic microcontroller, and of coordinating data transfers between the host and the microcontroller, including flow control, and error handling and retry mechanisms.

Detailed Description Text (73):
At this point, it suffices to briefly discuss the organization of data within the dual port RAM device 214. Referring now to FIG. 9, a memory map 300, which is the default memory map (or organization) of information stored in the dual port RAM device 214. It should be noted that the default memory map 300 is designed to support the Cy123 ISDN Controller device, disclosed in U.S. Pat. No. 5,541,930.

CLAIMS:

20. A RAM-based interrupt-driven serial interface device for establishing a communication link between a high performance serial bus host and a microcontroller device providing a control function, the interface device being operative to receive digital information in the form of information packets from the host and to process the packets and store the processed digital information in RAM memory buffers, then to generate an interrupt signal to the microcontroller device, and in response thereto, the microcontroller device being operative to access digital information stored in particular locations in the RAM memory buffers, and in response to said information, to write other information into other locations in the RAM memory buffers, then generate an interrupt signal to the interface device, which then interprets said other information, comprising:

timing and control means for controlling storage of data buffer descriptor information describing the particular RAM memory buffers;

dual port RAM means including data storage buffers, a descriptor storage buffer and means for permitting independent access to said descriptor storage buffer by both the microcontroller device and the serial interface device;

means by which the microcontroller device can signal said timing and control means that said descriptor storage buffer has been initialized;

hardware based storage means for storing descriptor information describing particular locations in said data storage buffers, such information to be used dynamically while transferring data to and from the high-performance serial bus and the data storage buffers described by the data buffer descriptor information, said timing and control means and said dual port RAM means being further operative to access said descriptor storage buffer and to copy the descriptor contents into said

hardware based storage means, and being further operative to manage said data transfer between the data storage buffers and the serial bus;

means for generating interrupt signals for signaling the microcontroller device upon completion of the data transfer between the serial bus and the data storage buffer described by said descriptor contents;

means for receiving address information and read and write strobes from the microcontroller device for accessing the described data storage buffer so that data can be exchanged between the dual port RAM means and the microcontroller device;

means for inhibiting storage of serial transfers from the host after an interrupt signal is sent to the microcontroller device thereby allowing the microcontroller to access uncorrupted data from the described data storage buffer;

means for receiving an interrupt signal from said microcontroller device signaling that access to the data storage buffer has been completed and that a data storage buffer described by the descriptor contents is available for use;

means for disabling said inhibit means so that additional data transfer between the high performance serial bus and the data storage buffer can occur;

means for detecting errors occurring during data transfers and for reinitializing said descriptor storage buffer for use in a "re-try" attempt; and

means for indicating that an error has occurred during data transfers and for signaling the host via the high performance serial bus that a "re-try" is required.

L2: Entry 12 of 26 File: USPT Jul 3, 2001

DOCUMENT-IDENTIFIER: US 6256723 B1
TITLE: Signal processing system with distributed uniform memory

Detailed Description Text (6):
The host 110 in one embodiment of the invention is a personal computer ("PC"), commonly known in the art. Such a system will generally include a host processor 121 as well as host memory 122, which may include long term memory such as a hard drive and short term memory such as RAM, where memory 122 may be internal or external to the host system 110. The host 110 is connected to the subsystem 114 via interface 112. In various embodiments of the invention, interface 112 can be a serial interface (e.g., RS-232), an ISA interface, a USB interface, a PCI interface, a PCMCIA interface, an LPC interface, or any of a plurality other interfaces as are generally known in the art.

Detailed Description Text (105):
Further, not only does each processor have read access to the cached information, but it can also write to the buffered windows of information, hence the buffered information must be updated. In order to keep the respective mapped memory locations updated, bits (in both the buffered locations and the regular processor memory space) that are changed are tagged. The ICCU monitors the tagged bits, and when bits are tagged, the ICCU requests core bus access. Upon an access grant, the ICCU initiates a sequence, e.g., a DMA transfer, to update the bits in the RISC memory space. Similarly, when buffered bits of DSP memory in the RISC memory space, or the 2K window buffered in the DSP memory space are written to by the RISC or other device, these bits are also tagged in the RISC memory space. The ICCU is notified, and will request access to the DSP memory resources to update the information. Thus, the processors write cycles do not need to be delayed to wait for access to the respective busses (DSP bus or core bus). In fact, by using such a memory mapping technique, the processors will have access to the memory spaces of other processors independent of timing requirements, load requirements or availability of memory space of the various memories and processors. Memory is thus shared in an autonomous and load balanced manner.

WEST

Generate Collection | Print

L2: Entry 12 of 26 File: USPT Jul 3, 2001

US-PAT-NO: 6256723
DOCUMENT-IDENTIFIER: US 6256723 B1

TITLE: Signal processing system with distributed uniform memory

DATE-ISSUED: July 3, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Hudson; Michael	Portland	OR		
Moore; Daniel L.	Vancouver	WA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Diamond Multimedia Systems, Inc.	San Jose	CA			02

APPL-NO: 09/ 060746 [PALM]
DATE FILED: April 15, 1998

INT-CL: [07] G06 F 12/06, G06 F 13/40

US-CL-ISSUED: 712/35; 712/36, 711/148, 711/129
US-CL-CURRENT: 712/35; 711/129, 711/148, 712/36

FIELD-OF-SEARCH: 711/149, 711/147, 711/114, 711/129, 711/131, 711/148, 711/130, 710/16, 707/205, 712/35, 712/36, 704/200, 704/269, 703/25

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4718080	January 1988	Serrano et al.	379/59
☐	4775931	October 1988	Dickie et al.	364/200
☐	4991085	February 1991	Pleva et al.	364/200
☐	5127041	June 1992	O'Sullivan	379/59
☐	5134648	July 1992	Hochfield et al.	379/98
☐	5167021	November 1992	Needham	395/275
☐	5181858	January 1993	Matz et al.	439/188
☐	5249218	September 1993	Sainton	379/59
☐	5302947	April 1994	Fuller et al.	340/825.34
☐	5333177	July 1994	Braitberg et al.	379/59
☐	5357625	October 1994	Arends	395/500
☐	5384890	January 1995	Anderson et al.	704/200
☐	5410667	April 1995	Belsan et al.	711/114
☐	5428671	June 1995	Dykes et al.	379/93
☐	5430793	July 1995	Ueltzen et al.	379/98
☐	5649001	July 1997	Thomas et al.	379/93.07

OTHER PUBLICATIONS

Competitive Analysis: Analog Devices, Inc.; ADSP-2104 vs. Texas Instruments TMS320C203 [ADSP-2104]; 1996 Analog Devices, Inc.

ART-UNIT: 213

PRIMARY-EXAMINER: Pan; Daniel H.

ATTY-AGENT-FIRM: Rosenberg; Gerald B. NewTechLaw

ABSTRACT:

A system is disclosed that includes a plurality of processors, which in some embodiments include DSPs and other microprocessors, and a distributed uniform memory. The distributed uniform memory is subdivided into a plurality of addressable memory spaces each of which are respectively primarily associated with one of the processors in the plurality of processors. At least an addressably contiguous portion of the addressable memory space primarily associated with one processor is mapped into the addressable memory space primarily associated with another processor. Thus, a processor will have access to the addressable memory space primarily associated with another processor, but will have such access independent of the load and timing requirements of the other processors.

21 Claims, 19 Drawing figures

L2: Entry 13 of 26 File: USPT Jun 19, 2001

DOCUMENT-IDENTIFIER: US 6249825 B1
TITLE: Universal serial bus interface system and method

Brief Summary Text (5):
When a peripheral device is first connected to the USB and the host computer through a standard USB communications port, the presence of the connected peripheral device is detected and a configuration process of the USB for the connected peripheral device, known as device enumeration, begins. The enumeration process assigns a unique USB address to the connected peripheral device, queries the connected peripheral device about its requirements and capabilities, writes data about the connected peripheral device into the host computer's operating system, and loads the appropriate software device driver from a storage location into the host computer's operating system. During the query, a data table stored in the peripheral device, which contains the particular peripheral device's configuration information, is read from the peripheral device into the host computer's memory. Upon completion of the enumeration process, the connected peripheral device is recognized by the host computer's operating system and may be used by application software being executed by the microprocessor of the host computer. The association of the device with the software device driver cannot be subsequently changed.

Brief Summary Text (6):
In a serial bus system, such as the USB, the only opportunity for associating software device drivers with a peripheral device is at the time when the peripheral device is plugged into the USB and the enumeration process occurs. Thus, to alter the configuration or personality of a peripheral device, such as downloading new code or configuration information into the memory of the peripheral device, the host computer system must detect a peripheral device connection or a disconnection and then a reconnection.

Brief Summary Text (12):
In addition, the universal serial bus interface system and method may be a single semiconductor chip which may be incorporated into a plurality of peripheral devices made by a plurality of manufacturers. The chip may initially have a generic configuration (e.g., not specific to a particular peripheral device). Then, the appropriate configuration information for a particular peripheral device and manufacturer may be downloaded to the chip, an electronic simulation of the disconnection and reconnection of the peripheral device occurs, the peripheral device is recognized as a new, manufacturer specific peripheral device and the appropriate software device driver is loaded into the memory of the host computer.

Detailed Description Text (3):
FIG. 1 is a diagram illustrating a standardized bus interface, such as a conventional computer system 20, that may include a host computer system 22 and a peripheral device 24. The peripheral device is connected to the host computer by a universal serial bus (USB) 26. The host computer may include a central processing unit (CPU) 28 connected to a USB interface (I/F) circuit 30, and the USB standard provides a universal electrical and physical interface for the peripheral devices via bus 26. The CPU executes software application code located in a memory 31 and communicates data to and from the peripheral device through the USB interface and the USB 26. The host computer may also include an operating system 32 which may include a software device driver 33. The peripheral device 24 may include a USB interface circuit 34, a CPU 36 and a non-volatile memory 38 that may store

configuration information describing the characteristics of the peripheral device. The non-volatile memory may be a read only memory (ROM) or an erasable programmable read only memory (EPROM).

Detailed Description Text (4):
When the peripheral device is initially connected to the USB, an enumeration process is conducted in which the host computer determines the characteristics of the peripheral device by receiving the configuration information from the memory 38 within the peripheral device, and configures the USB according to the characteristics of the peripheral device. As shown, the configuration information about the characteristics of the peripheral device in a conventional USB system is stored in a non-volatile memory 38 on the peripheral device. The data about the characteristics of the peripheral device is programmed into the non-volatile memory at the factory, and the characteristics of the peripheral device may not be easily altered. In addition, the memory in the peripheral device stores all of the configuration information about the peripheral device which may require a large amount of memory in the peripheral device.

Detailed Description Text (6):
FIG. 2 is a diagram illustrating a computer system 50 that may have a universal serial bus system in accordance with the invention. The computer system may include a host computer 52 connected to a peripheral device 54 by a universal serial bus (USB) 60. The host computer may include a CPU 62, a memory 64, an operating system 65 and a USB interface circuit 66. One or more peripheral device drivers, such as a first peripheral device driver 68, may be stored in the operating system 65. Each device driver contains information about the proper configuration of the USB for a particular class of peripheral devices. The operating system within the host computer may also contain a plurality of different configuration information sets 70, which may include configuration data for a particular peripheral device (including which device driver to use), microprocessor code to be executed by a CPU located in the peripheral device, or logic configuration data to configure logic circuits in the peripheral device. This invention advantageously enables these configuration information sets to be updated or altered easily since they are located in the host computer and not in a non-volatile memory in the peripheral device.

Detailed Description Text (9):
In operation, during the initial factory configuration of the peripheral device with the USB interface system in accordance with the invention, the memory may store an identification code indicating the appropriate configuration information set to be loaded. Thus, when the peripheral device is first connected to the USB, the configuration information 70, including any microprocessor code applicable to the peripheral device and the appropriate configuration data for the peripheral device may be downloaded over the USB into the memory 74 of the peripheral device 54 as shown by the dashed arrow 78. The electrical simulation of the disconnection and reconnection of the peripheral device from the USB, as described below, may be initiated and a re-enumeration process may occur. During the re-enumeration process, the newly downloaded configuration information may be used to reconfigure the USB for the peripheral device and the host computer may select the appropriate software device driver 68 for the peripheral device based on the configuration information and load the device driver into memory 64 as shown by arrow 80. For example, a plurality of different peripheral devices manufactured by different companies may each include a USB interface system in accordance with the invention. The USB interface system for each peripheral device is identical (e.g. has a USB interface circuit and a memory) except that each memory may contain an identification code that is unique to, for example, a particular manufacturer. When one of the peripheral devices is connected to the USB and the host computer, the appropriate configuration information for the peripheral device, based on the identification code, is downloaded over the USB to the memory of the peripheral device and the appropriate software device driver is loaded into the memory of the host computer. Thus, a plurality of different peripheral device may include the same USB interface system hardware since the configuration information is located in the operating system of the host computer. Now, a conventional USB interface circuit and the universal serial bus interface circuit in accordance with the invention that permits the electronic disconnection and reconnection of the peripheral device will be

described.

Detailed Description Text (14):
FIGS. 5, 6, and 7 are diagrams illustrating three different peripheral devices from different manufacturers, for example, being connected to a host computer in accordance with the invention. In each Figure, a computer system 140 may include a host computer 142, a plurality of peripheral devices, such as peripheral device "A" 144 (shown in FIG. 5), peripheral device "B" 146 (shown in FIG. 6), peripheral device "C" 148 (shown in FIG. 7) and a USB bus 149. The host computer 142 may include a CPU 150, a memory 152, an operating system 154 and a USB interface circuit 156. The operating system, in this example, may include a plurality of software device drivers, such as device driver "A" 158, device driver "B" 160 and device driver "C" 162, and a plurality of configuration information sets, such as device "A" characteristics 164, device "B" characteristics 166 and device "C" characteristics 168.

Detailed Description Text (16):
As shown in FIG. 5, peripheral device "A" 144 may have a unique manufacturer signature in the non-volatile memory 178. When the peripheral device is connected to the computer system, the enumeration process begins in which the USB interface system 170 is recognized by the USB as a generic device and the unique manufacturer signature is read from the non-volatile memory by the CPU 150 over the USB 149. The unique signature identifies device "A" characteristics 164 as the appropriate configuration information and that configuration information may be downloaded over the USB 149 into the memory 174 of the peripheral device as shown by dashed arrow 180. Then the electrical simulation of the disconnection and reconnection of the peripheral device occurs, as described above, which cause re-enumeration of the peripheral device. During re-enumeration, device driver "A" 158, which is identified by device "A" characteristics 164 as the appropriate device driver, is loaded from the operating system into the memory, as shown by arrow 182, such that the peripheral device is now recognized as a peripheral device with device "A" characteristics. Thus, a generic hardware USB interface system may be incorporated into a peripheral device and the particular characteristics for the particular peripheral device may be later downloaded from the host computer into the peripheral device.

Detailed Description Text (17):
Similarly, as shown in FIGS. 6 and 7, the peripheral devices 146, 148 may include the generic USB interface system and a unique manufacturer signature in the non-volatile memory and may be re-enumerated in accordance with the invention so that the appropriate device characteristics are downloaded from the host computer over the USB into the memory of the peripheral device and the appropriate device driver may be selected by the host computer. Thus, peripheral device "B" 146 (shown in FIG. 6) may have device "B" characteristics 166 downloaded into its memory, as shown by dashed arrow 184 in FIG. 6, and the host computer may use device driver "B" 160, as shown by arrow 186 in FIG. 6. The peripheral device "C" 148 (shown in FIG. 7) may have device "C" characteristics 168 downloaded into its memory as shown by the dashed arrow 188 in FIG. 7 and the host computer may use device driver "C" 162 as shown by arrow 190 in FIG. 7. Thus, a generic USB interface system may be incorporated into a plurality of different peripheral devices, the appropriate configuration information may be downloaded into the peripheral device, and the re-enumeration recognizes the peripheral device as a manufacturer specific device. The re-enumeration of the generic peripheral device ensures that the host computer discards all device driver information about the generic connection and loads the manufacturer-specific device driver software.

Detailed Description Text (19):
The universal serial bus interface system in accordance with the invention provides several advantages. The system provides an easy technique to associate new device driver software with a peripheral device, known as re-enumeration. A peripheral device may be disconnected and reconnected to the USB without the physical disconnection and reconnection of the peripheral device to cause re-enumeration to occur. In addition, because the peripheral device is not physically disconnected from the host computer, the peripheral device may use the electrical power available over the USB bus to maintain the configuration information if it is in a volatile

memory and to perform tasks during the simulated disconnection. The characteristics of the peripheral devices contained in one or more configuration information sets may be stored in the host computer so that the configuration information may be easily changed. The combination of the configuration information sets stored in the host computer and the electronic disconnection and reconnection of the peripheral device permits the characteristics of a peripheral device to be changed rapidly without physical disconnection of the peripheral device. In addition, the configuration of any peripheral device connected to the USB may be altered or changed multiple times. The system also permits a generic USB interface system to be incorporated into a plurality of peripheral devices and then each peripheral device to be configured with manufacturer specific configuration information.

WEST

Generate Collection | Print

L2: Entry 13 of 26 File: USPT Jun 19, 2001

US-PAT-NO: 6249825
DOCUMENT-IDENTIFIER: US 6249825 B1

TITLE: Universal serial bus interface system and method

DATE-ISSUED: June 19, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Sartore; Ronald H.	San Diego	CA		
Larky; Steven P.	Del Mar	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Cypress Semiconductor	San Jose	CA			02

APPL-NO: 09/ 476923 [PALM]
DATE FILED: January 4, 2000

PARENT-CASE:
This is a continuation of U.S. Ser. No. 08/886,923, filed Jul. 2, 1997 now U.S. Pat. No. 6,012,103.

INT-CL: [07] G06 F 13/368

US-CL-ISSUED: 710/8; 710/9, 710/10
US-CL-CURRENT: 710/8; 710/10, 710/9

FIELD-OF-SEARCH: 713/1, 713/100, 710/8, 710/7, 710/9, 710/10, 320/21, 709/220, 709/221, 709/250, 711/103-105

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4641261	February 1987	Dwyer et al.	364/900
☐	4862355	August 1989	Newman et al.	364/200
☐	5289580	February 1994	Latif et al.	395/275
☐	5428748	June 1995	Davidson et al.	395/275
☐	5488657	January 1996	Lynn et al.	379/395
☐	5497067	March 1996	Shaw	320/21
☐	5574859	November 1996	Yeh	395/200.1
☐	5577213	November 1996	Avery et al.	395/280
☐	5586268	December 1996	Chen et al.	395/250
☐	5598409	January 1997	Madonna et al.	370/364
☐	5606672	February 1997	Wade	395/308
☐	5615344	March 1997	Corder	395/309
☐	5634074	May 1997	Devon et al.	395/828
☐	5655148	August 1997	Richman et al.	370/94
☐	5671355	September 1997	Collins	709/250
☐	5673031	September 1997	Meier	340/825.08
☐	5687346	November 1997	Shinohara	711/130
☐	5701429	December 1997	Legvold et al.	711/114
☐	5794033	August 1998	Aldebert et al.	713/100
☐	5802558	September 1998	Pierce	711/115
☐	5838907	November 1998	Hansen	709/220
☐	5898861	April 1999	Emerson et al.	710/7
☐	5982879	November 1999	Lucey	379/389
☐	6012103	January 2000	Sartore et al.	710/8
☐	6038667	March 2000	Helbig, Sr.	713/200
☐	6049870	April 2000	Greaves	713/1

ART-UNIT: 212

PRIMARY-EXAMINER: Lee; Thomas

ASSISTANT-EXAMINER: Du; Thuan

ATTY-AGENT-FIRM: Maiorana, P.C.; Christopher P.

ABSTRACT:

A system for reconfiguring a peripheral device having a first configuration connected by a computer bus and a port to a host computer. The system comprises a first circuit and a second circuit. The first circuit may be configured to download information for a second configuration from the host computer into the peripheral device over the computer bus. The second circuit may be configured to electronically simulate, over the computer bus, a physical disconnection and reconnection of the peripheral device to reconfigure the peripheral device to the second configuration.

20 Claims, 8 Drawing figures

First Hit Fwd Refs

Generate Collection Print

L8: Entry 9 of 10 File: USPT Feb 20, 2001

DOCUMENT-IDENTIFIER: US 6192420 B1
TITLE: Controller with a vendor/product identification pin for a Universal Serial Bus by various manufacturers

Abstract Text (1):
A Universal Serial Bus (USB) controller with a VIDS pin to set up a plurality of vendor identification which can be accompanied with interface and software program. Thus, one chip can achieve Vendor ID/Product ID (VID/PID) of 2.sup.N types (N is a positive integer equal to or greater than 1) in accordance with various USB manufactures and products.

Brief Summary Text (6):
In view of the above problem, well-known manufacturers including Microsoft, Intel, IBM, DEC, Compaq, NEC, Nortel, and other international companies, have jointly set up an open type framework organization to establish the standards for the USB, with the issue of edition 1.0 of its specifications in January 1996. These specifications are applicable to personal computers with a PC framework. Microsoft has adhered to the USB as the standard for performance in the new edition of their Windows operation system, Windows 98. In addition, major manufacturers of the mother board have included USB interfaces in their products.

Brief Summary Text (7):
In response to the above specifications, manufacturers of the known art have provided their products with various types of USB controllers, such as the USB keyboard controller, USB mouse controller, USB hub controller, etc.

Brief Summary Text (9):
According to USB specifications, in order for the peripheral output/input equipment of a PC system to be provided a function of plug and play as the goal, each function should be provided with VID/PID for identification by the PC. Since VID from different manufacturers of USB products are not identical (for example, company A and company B produce compatible USB keyboards which in different VID), even the same series of products by the same brand may have different PID (such as in USB keyboards, where the scanning method and language may differ and lead to different PID). The design used in this invention allows for the initial assembling of 2.sup.N types (N.gtoreq.1) of VID/PID, and then uses VIDS pins in conjunction with a software program and interface control to select the setting/reading of the VID/PID. The features of this design are not defined in edition 1.0 of the USB specifications. The addition of the function will make the design more versatile in meeting the demands of manufacturers who use the USB controller when producing computer peripheral equipment which have USB functions (for example, in the production of a USB keyboard, USB mouse, and USB hub controller). In order to meet the needs of general USB controllers, the present invention requires all USB product designs and manufacturers to provide only one kind of product for every usage. This will significantly lower inventory risks and production costs.

Detailed Description Text (2):
FIG. 1 is the block diagram of a Universal Serial Bus (`USB") controller (11) of the present invention. The controller comprises a micro-controller (12), a USB module(13), an input bus (14), and a VIDS pin as an output pin to set up a

h e b b cg b cc e

plurality of vendor and product identification code. The micro-controller (12) contains a data memory (121) and non-volatile memory (122) for recording VID/PID from various manufacturers and products, and other relevant information.

Detailed Description Text (3):
FIG. 2 shows a preferred embodiment of the USB controller of the present invention. The terminals 1,2, . . . N of the USB controller (21) are connected via resistors R1, R2, . . . RN, to a higher voltage level (in this embodiment, +5V). Under these conditions, the terminals 1, 2, . . . N, are pulled high. At the same time, terminals 1,2 . . . N are connected through the interface (22) (in this case formed from diodes D1, D2, . . . DN) to the VIDS pin which serve as an output pin. In this embodiment, under ordinary conditions, the output voltage level of VIDS pin is high. Only on the setting/reading values of the VID/PID in the USB controller, the output voltage level is changed to a low level. Thus, the manufacturer or designer can use the installation of diodes D1, D2, . . . DN to obtain 2.sup.N combinations. Once the VIDS is at a low voltage level, the input terminals with diodes can be set at a low level. Input terminals without being installed diodes maintain a high voltage level that is originally pulled high. In other words, by setting the VIDS pin at a low voltage level with or without the diodes D1, D2, . . . DN installed to set the values of input terminals 1, 2, . . . N, one of 2.sup.N type combinations (decided by the diodes D1, D2, . . . DN) is obtained.

US Reference Patent Number (9):
6012103

First Hit Fwd Refs

☐ Generate Collection | Print

L8: Entry 9 of 10 File: USPT Feb 20, 2001

US-PAT-NO: 6192420
DOCUMENT-IDENTIFIER: US 6192420 B1

TITLE: Controller with a vendor/product identification pin for a Universal Serial Bus by various manufacturers

DATE-ISSUED: February 20, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Tsai; Chuan-Ching	Taipei			TW
Gee; Joseph	Taipei			TW
Wu; Chun-Min	Taipei Hsien			TW
Liao; Chen-Yuan	Tao Yuan Hsien			TW

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Weltrend Semiconductor Inc.	Hsin-Chu			TW	03

APPL-NO: 09/ 119377 [PALM]
DATE FILED: July 20, 1998

FOREIGN-APPL-PRIORITY-DATA:

COUNTRY	APPL-NO	APPL-DATE
TW	86213537	August 9, 1997

INT-CL: [07] G06 F 13/00

US-CL-ISSUED: 710/10; 710/8, 710/62, 710/63, 710/104
US-CL-CURRENT: 710/10; 710/104, 710/62, 710/63, 710/8

FIELD-OF-SEARCH: 710/8, 710/9, 710/10, 710/104, 710/62, 710/63, 395/500, 395/861, 395/882

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL | Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4773005	September 1988	Sullivan	710/9
☐	5317693	May 1994	Cuenod et al.	710/9

h e b b g e e e f c e c e ge

☐	5404460	April 1995	Thomsen et al.	710/9
☐	5774114	June 1998	Suzuki	345/157
☐	5860028	January 1999	Pecore	395/861
☐	5905370	May 1999	Bryson	323/283
☐	5987548	November 1999	Dixon	710/104
☐	5991546	November 1999	Chan et al.	395/882
☐	6012103	January 2000	Sartore et al.	710/8

ART-UNIT: 272

PRIMARY-EXAMINER: Lee; Thomas C.

ASSISTANT-EXAMINER: Peyton; Tammara

ATTY-AGENT-FIRM: Christensen O'Connor Johnson Kindness PLLC

ABSTRACT:

A Universal Serial Bus (USB) controller with a VIDS pin to set up a plurality of vendor identification which can be accompanied with interface and software program. Thus, one chip can achieve Vendor ID/Product ID (VID/PID) of 2.sup.N types (N is a positive integer equal to or greater than 1) in accordance with various USB manufactures and products.

4 Claims, 5 Drawing figures

First Hit Fwd Refs

End of Result Set

☐ Generate Collection Print

L1: Entry 5 of 5 File: USPT Dec 7, 1999

DOCUMENT-IDENTIFIER: US 6000042 A
TITLE: Fault detection on a dual supply system for a universal serial bus system

Detailed Description Text (2):
FIG. 1a shows the architecture of a universal serial bus repeater 5 which includes a universal serial bus controller 10 connected to a number of outside ports. The universal serial bus controller 10 controls the routing of the signals from the upstream port to the downstream ports and from the downstream ports to the upstream port and error detection and recovery. Universal Serial Bus (USB) Port 0 (12) is the universal serial bus upstream port which is typically connected to a host computer. Universal Serial Bus Ports 1 to 7 (18) are universal serial bus downstream ports which are typically connected to universal serial bus devices. D+ and D- signify signals which are sent to and from both the upstream and downstream ports. At the connection to the universal serial bus port 0 (12), there is a 5 Volt bus voltage signified as VBUSIN. The VBUSIN voltage is input to the 5 V to 3.3 V Regulator 14, with the output of the regulator being sent to power the universal serial bus controller 10, the fault detect circuit 24, and the power-on reset circuit 17 (i.e., resetting the controller during power-up) (shown in FIG. 5). The universal serial bus controller 10 also has a crystal input (XTAL) 16 which operates at 48 MHz for purposes of timing.

First Hit Fwd Refs
End of Result Set

☐ Generate Collection Print

L1: Entry 5 of 5 File: USPT Dec 7, 1999

US-PAT-NO: 6000042
DOCUMENT-IDENTIFIER: US 6000042 A

TITLE: Fault detection on a dual supply system for a universal serial bus system

DATE-ISSUED: December 7, 1999

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Henrie; James B.	Grayslake	IL		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
3Com Corporation	Santa Clara	CA			02

APPL-NO: 08/ 918013 [PALM]
DATE FILED: August 25, 1997

INT-CL: [06] G06 F 11/22

US-CL-ISSUED: 714/40; 713/340
US-CL-CURRENT: 714/40; 713/340

FIELD-OF-SEARCH: 714/14, 714/22, 714/43, 714/40, 713/340, 713/300

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5675813	October 1997	Holmdahl	
☐	5884086	March 1999	Amoni et al.	713/300

OTHER PUBLICATIONS

"Micrel MIC2525 USB High-Side Power Switch Advance Information" (4 sheets).
"Micrel MIC2526 Dual USB High-Side Power Switch Advance Information" (6 sheets).
Solectron Texas Logic Diagram TUSBK7HUB (5 sheets).
Solectron Texas Logic Diagram TUSBK4HUB (4 sheets).
Solectron Texas Blueprint 9806020-001 C (2 sheets).
Solectron Texas Logic Diagram (5 sheets); Jul. 17, 1997.

h e b b g e e e f c e e ge

Texas Instruments Manual entitled "SN75240 Dual Universal Serial Bus Port Transient Suppressor;" pp. 1-4; Feb. 1997.
Texas Instruments Manual entitled "TPS7101Q, TPS7133Q, TPS7148Q, TPS7150Q, TPS7101Y, TPS7133Y, TPS7148Y, TPS7150Y Low-Droput Voltage Regulators;" pp. 1-14, 18-32; Jan. 1997.
Texas Instruments Manual entitled "TPS2010, TPS2011, TPS2012, TPS2013, TPS2010Y Power-Distribution Switches;" pp. 6-1 through 6-18; Aug. 1995.
Solectron Texas Blueprints (9 sheets); 1996-1997.
Texas Instruments Manual entitled "TUSB2140 Data Manual--4-Port Hub With an Embedded Function for the Universal Serial Bus;" pp. iii through B-2 (63 sheets); Nov. 1997.
Texas Instruments Manual entitled "TUSB2040 4-Port Hub for the Universal Serial Bus;" pp. 1-13; Aug. 1997.
Texas Instruments Manual entitled "TUSB2070 7-Port Hub for the Universal Serial Bus;" pp. 1-12; Aug. 1997.
Intel brochure regarding USB Family Evaluation Board (2 sheets); 1998.
Intel Manual entitled "8x931HA USB Customer Hub Preliminary Datasheet;" pp. 1-12; Nov. 1997.
Intel Manual entitled "Advance Information--8x930Hx Universal Serial Bus Hub Peripheral Controller;" pp. iii-33; May 1997.
Cherry Electrical Products brochure regarding G80-3000HXMUS Keyboard; Aug. 29, 1997.
Belkin brochure regarding Express Bus.TM. Hub; undated.
CATC brochure regarding Andromeda.TM. USB Hub and Orion.TM. self-powered USB Hub; (2 sheets); undated.
National Semiconductor Manual entitled "USBN9602 (Universal Serial Bus) Full Speed Function Controller With DMA Support;" pp. 1-43; Nov. 1997.
Manual entitled "Universal Serial Bus Specification" Revision 1.0; pp. 1-10, 217-268; Jan. 15, 1996.
Texas Instruments Manual entitled "TPS2014, TPS2015 Power Distribution Switches;" pp. 1-21; Aug. 1997.
Solectron Texas Blueprints (9 sheets); 1996-1997.
Texas Instruments Manual entitled "(EVM) Evaluation Module Documentation--TUSBEM 2140 (USB 4-Port Hub & I.sup.2 C Interface)" (11 sheets); Aug. 1, 1997.
Logic Diagram for TUSB2040 (4 sheets).
Logic Diagram for TUSB2070 (5 sheets).

ART-UNIT: 277

PRIMARY-EXAMINER: Heckler; Thomas M.

ATTY-AGENT-FIRM: McDonnell Boehnen Hulbert & Berghoff

ABSTRACT:

A method and apparatus for a dual power supply on a universal serial bus system using an overcurrent detect circuit. Dual power supplies in the universal serial bus system allows for greater flexibility of operation and is based on two separate power systems. The first power system is achieved using the power line on the bus connecting to the universal serial bus controller. The second power system is a separate power supply to power the downstream ports. Moreover, the universal serial bus system has an overcurrent and thermal error detect circuit based on the power system in order to achieve an efficient and cost effective method and apparatus in which to notify the universal serial bus controller of any error in the power system.

47 Claims, 14 Drawing figures